UNIVERSITÉ DE MONTRÉAL

CENTRE DE RECHERCHE SUR LES TRANSPORTS

CASE POSTALE 6128, SUCCURSALE A, MONTRÉAL, QUÉBEC, H3C 3J7

ENHANCEMENTS TO THE HASTUS CREW SCHEDULING ALGORITHM

by

Jean-Marc Rousseau*, Réjean Lessard*
and Jean-Yves Blais**

* Centre de recherche sur les transports, Université de Montréal

** GIRO, Inc., Montréal, Québec, CANADA

Cette étude a été publiée grâce à une subvention du fonds F.C.A.R. pour l'aide et le soutien à la recherche.

Université de Montréal
Centre de recherche sur les transports - Publication #321
Juin 1983

Dépôt légal - 1er trimestre 1987

ABSTRACT

The purpose of this paper is to update the description of the HASTUS algorithm presented at the Leeds Workshop in the light of developments since the Leeds Conference. While some of the improvements have been already implemented (e.g. the matching algorithm), other improvements described here are still under research and development. The basic approach however has remained unchanged. For the sake of completeness of the current paper we borrow heavily from our previous paper [8] to describe the general strategy that we are currently using.

KEY WORDS : crew scheduling, transit, matching

RÉSUMÉ

Dans ce document nous présentons les diverses modifications et améliorations apportées au logiciel HASTUS depuis la conférence de Leeds. Plusieurs de ces modifications ont déjà été implantées (algorithme de couplage, par exemple) tandis que d'autres font encore l'objet de recherches. L'approche de base reste cependant la même et pour que ce document soit complet en lui-même, nous reprenons d'un document précédent [8] la description de la stratégie générale.

MOTS CLÉS : horaire de personnel, transport collectif, couplage

THE PROBLEM

The bus driver scheduling (BDS) problem in a transit company involves establishing, at minimum cost for each day of the week, a list of workdays which assign a driver to each bus in the timetable and respect all clauses in the union contract. In the approach discussed here it is assumed that the bus schedule is known and that once the list of feasible workdays is established, the problem is solved. In fact, in most North-American companies, the assignment of workdays to drivers is carried out by the drivers themselves and this selection is done on a seniority basis.

The difficulty of the problem arises directly from the kind of service that a transit company must offer and the travel patterns of a population. The number of vehicles in service may be much greater at peak hours than at off-peak hours. This obviously necessitates either part-time drivers, or split-shift workdays for full time drivers, or both. In most companies, unions refuse or severely restrict the part-time driver solution. Several rules have also appeared defining legal split-shift workdays and working conditions which limit the number of and/or compensate the drivers for less desirable workdays. These working conditions are described in more detail in several papers [11,13].

We recall here the basic terminology and some related rules which characterize the problem.

A <u>block</u> is the itinerary of a vehicle between its departure from and its return to the garage. It includes all deadhead time required to take the bus in and out of the garage and to and from the route(s) it services. There are generally short blocks to cover the peak periods and long blocks for the basic service.

Relief times are the times corresponding to points on the route where a change of drivers is possible. In general the number of such points is small.

A workday is a daily assignment consisting of one or more pieces of work, which must satisfy the union contract rules.

A piece of work (or piece) refers to the period of time during which a driver works continuously with the same vehicle without a break. Generally the number of pieces of work in a workday is limited (2, 4 or 5) and the pieces must have a limited duration.

A tripper is a small piece of work which is normally done in overtime by regular drivers or assigned to stand-by drivers. In general the companies hope for a few or no trippers. By extension and simplicity of notation we consider that a tripper is a workday. However we assume that there is no explicit upper bound on the number of trippers thus ensuring the existence of a feasible schedule. Moreover, a high penalty is imposed on trippers.

A block partition is a set of pieces of work which exactly covers the block. The BDS problem has been described in detail and several approximate solution methods have been proposed. The proceedings from the three workshops on the BDS problem provide an excellent set of references [14,15,16].

In the next section we recall the general mathematical programming formulation of the BDS problem that we are using. Because in general this problem is too large, we introduce a relaxation of the problem and present a solution approach.

THE MODEL

For simplicity of presentation, we now assume that there are at most two pieces in a workday. The extension to three or more pieces of work is simple. In this section a general formulation of the problem is introduced. This generality is restricted only by the fact that we assume that the minimum break required by the union in a workday is always sufficient to allow for driver travel time between two pieces of work and that there is no cost associated with the travel time. If this is not the case, our solution strategy is not modified; but this fact is taken into account when the pieces of work are matched together to form workdays. Reference [1] contains a generalization of this formulation that allows the consideration of these travel times.

The notation required for the HASTUS model is first introduced. We denote by a pair (i,j) a piece of work starting at time i and ending at time j. Only feasible pairs (i,j) are considered, that is pairs such that both times are included in a given block. Note that (i,j) may be feasible relative to several blocks. For practical reasons, we also consider in the feasible set of pieces the null piece (0,0).

A quadruplet (i,j,k,h) denotes a workday made up of the feasible pairs (i,j) and (k,h). Only workdays (i,j,k,h) which are compatible with the union contract are considered. If either (i,j) or (k,h) is a null piece, the workday either has no break or corresponds to a tripper.

In addition we define:

L : the number of blocks.
K : the number of distinct pieces.
I : the set of feasible quadruplets (i,j,k,h).
$I(i,j) \subset I$: the set of quadruplets where one of the pieces is (i,j).

T_ℓ : the set of all times k which are either relief times, starting times or ending times for block ℓ.

x_{ijkh} : an integer non-negative variable corresponding to the number of drivers assigned to a workday (i,j,k,h).

x : the vector with components x_{ijkh}, $(i,j,k,h)\in I$.

c_{ijkh} : the cost of the workday (i,j,k,h); this is normally the direct cost according to the union contract.

y^ℓ_{ij} : a binary variable taking value 1 if piece (i,j) is used on a block ℓ as part of a driver workday.

y^ℓ : the vector with components y^ℓ_{ij} for <u>all</u> feasible (i,j).

y : the vector with components y^ℓ, $\ell=1,.,L$.

Y : the set of all feasible y such that $y^\ell_{ij}\ell=0$ if $i\in T_\ell$ or $j\in T_\ell$, $\ell=1,2,.,L$.

The problem can now be formulated as follows [3] :

$$\text{Min} \sum_I c_{ijkh}\, x_{ijkh}$$

s.t. i) $\sum_{I(i,j)} x_{mnkh} - \sum_\ell y^\ell_{ij} = 0$ feasible (i,j) (1)

ii) $Dx \geq d$

iii) $\sum_{i\in T_\ell} y^\ell_{ik} - \sum_{j\in T_\ell} y^\ell_{kj} = b^\ell_k$ for all $k\in T_\ell$, ℓ,

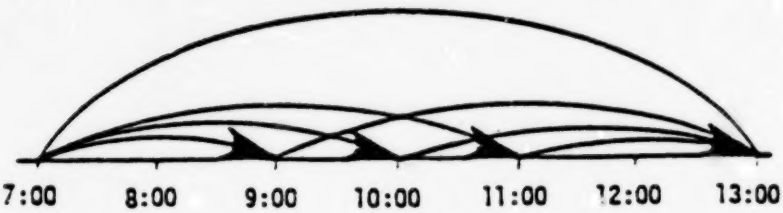
where $b^\ell_k = \begin{cases} -1 \text{ if k is the starting time of block } \ell \\ +1 \text{ if k is the ending time of block } \ell \\ 0 \text{ otherwise} \end{cases}$

iv) $x \geq 0$ and integer
$y \in Y$

The constraints (iii) correspond to the flow formulation used to partition each block ℓ into pieces of work. Figure 1 illustrates the concept. The feasible pieces of work (we are assuming here a minimum length of two hours) are represented by arcs and listed in the figure. We have assumed for simplicity that a relief point exists every hour in this example. The indices k in the constraint formulation correspond to points where a change of drivers is feasible and the constraints ensure that a

flow of one goes from the origin of the block (7:00) to its end (13:00), using arcs (i,j) corresponding to feasible pieces. Constraints (iii) define a flow problem on an uncapacitated network.

By constraints (i) we ensure that any feasible piece (i,j), used to partition a block will be used in a workday of type (i,j,k,h,) or (k,h,i,j,).



feasible pieces are

7 - 9	7 - 11	9 - 11	10 - 13
7 - 10	7 - 13	9 - 13	11 - 13

Figure 1

The flow formulation for the partition of a block

Constraints of type (ii) refer to any special constraints which may arise in a real life problem. Most of the time, these constraints take into account particular clauses of the union contracts like a bound (either lower or upper) on the number of workdays without break or a bound on the average length of the workdays.

Unless the problem is small (i.e. the number of blocks is small and thus the number of pieces and workdays is not too large), this formulation seems to be impractical. Given a medium-sized problem like that of Québec City or Florence, we can easily generate over five thousand pieces and one million workday variables without considering the flow variables; and moreover, there is the diffi- ty of solving the x_{ijkh} in integer form. In fact, this formulation is nearly culty equivalent to the set covering formulation. However, formulation (1) is more amenable to a solution

strategy that can handle very large problems. The problem can be stated in a more compact form as :

$$\begin{aligned} &\text{Min} \quad cx \\ &\text{i)} \quad B\,x - \sum_{\ell} y^{\ell} = 0 \\ &\text{ii)} \quad D\,x \qquad\qquad \geq d \qquad\qquad (2) \\ &\text{iii)} \quad E\,y^{\ell} = b^{\ell} \qquad\qquad \ell=1, \ldots, L \\ &\text{iv)} \quad x \geq 0, \quad y \in Y \end{aligned}$$

The definition of c, B, E and b^{ℓ} follows from formulation (1) of the problem.

Solution strategy

The chosen solution strategy is to use the obvious decomposition of the problem into the generation of a partition of each block into pieces of work (constraint (iii)) and the matching problem (constraint (i)) to form workdays. Constraints (ii) also have to be satisfied. Four main steps compose this strategy :

1. Using a relaxation of problem (1) the HASTUS-macro linear program is derived; this indicates the types of pieces of work that should be cut and the types of workdays that should be generated to minimize the cost of the solution.
2. The blocks are partitioned into pieces so that they correspond as closely as possible to the HASTUS-macro optimal solution. At this step a quadratic integer program is solved heuristically.
3. A matching problem is set up with the pieces of work. This matching problem is solved using a flow algorithm. The optimal solution to the matching problem is obtained most of the time.
4. A marginal improvement algorithm is called to modify the block partitions and improve the solution. Marginal costs of the pieces of work

are derived from the matching problem and used to repartition the blocks.

A RELAXATION OF THE MODEL : THE HASTUS-MACRO APPROACH

First, the integrality of the x variables is relaxed, since several methods exist to derive reasonably good integer solutions when a continuous solution has been found. Secondly, we assume that the starting times, relief times and ending times for the blocks may only occur at predetermined times $t \varepsilon T$, for example every 15 or 30 minutes. In the latter case, this means that all bus blocks are approximated to the nearest half-hour, and relief points are possible at some of, or at each, half-hour period. More complicated schemes could also be devised; for example, one could use different time periods for peak and off-peak times. This relaxation of the problem considerably reduces the number of possible pieces of work (i,j); however, it is important to note that a piece (i,j) may now be feasible for several blocks, which also means that x_{ijkh} may be greater than 1. All i,j,k,h are now in T.

Moreover, the problem is further relaxed by requiring that the workdays selected be sufficient to cover the total requirement of drivers per time period (i.e. from one predetermined time to the next), instead of requiring that they exactly cover all the blocks individually. Using the same notation, the relaxed problem can be written as follows :

$$\text{Min} \sum_{I} c_{ijkh}\, x_{ijkh}$$

s.t.

$$\text{i)} \sum_{I(t)} x_{ijkh} \geq N_t \qquad \text{for all } t \varepsilon T$$

$$\text{ii)} \quad Dx \geq d \qquad (3)$$

$$\text{iii)} \sum_{I(pq)} x_{ijkh} \geq Q_{pq} \qquad \text{for all (pq) such that a small block exists from p to q } (p,q \varepsilon T)$$

$$\text{iv)} \sum_{L(t)} x_{ijkh} \geq K_t \qquad \text{for all } t \in T$$

$$\text{v)} \; x_{ijkh} \geq 0 \quad \text{(integrality is relaxed)}, \; i,j,k,h \in T,$$

where :

T : the set of predetermined times which could be relief times, starting or ending times of blocks.

$I(t)$: the set of workdays $(i,j,k,h) \in I$ such that $i < t < j$ or $k < t < h$

$L(t)$: the set of workdays $(i,j,k,h) \in I$ with a piece starting at time t $(i = t$ or $k = t)$.

N_t : the number of buses in operation during the time period starting at t.

Q_{pq} : the number of blocks from p to q.

K_t : the number of blocks starting at t.

The set of constraints (i) ensures that during all periods of the day the number of drivers working is greater than or equal to the number of vehicles in circulation. Constraint (ii) refers to union contract constraints as previously prescribed. In constraint (iii), the number of pieces of work from p to q is at least as large as q_{pq}, the number of small blocks from p to q. A small block is defined by the user as a block that cannot be partitioned and should be allocated as one piece of work to a driver. This generally corresponds to blocks with a duration of less than twice the minimum duration of a piece of work.

In constraint (iv), the number of pieces of work beginning at t must at least be equal to K_t, the number of blocks starting at t. Finally, x_{ijkh} is a continuous variable that can take any positive value in this relaxation. However, for x_{ijkh} to exist there must be a piece (i,j) and a piece (k,h), each feasible with respect to at least one block.

The HASTUS-macro approach is independently described in another paper [2], and has been used on several occasions to analyse modifications to the drivers' union contract.

Partitioning the blocks

In the HASTUS system, the relaxation (HASTUS-macro) described earlier is used to help generate a first feasible solution as close as possible to the lower bound indicated. This is done first by generating an initial block partition that uses similar types of pieces of work and in approximately the same number as indicated by HASTUS-macro (3).

The following problem is considered :

$$\text{Min } (B\bar{x} - \sum_{\ell}(y^{\ell}))^2$$

$$E\, y^{\ell} = b^{\ell} \qquad \ell=1,\ldots,L \qquad (4)$$

$$\text{where } (w)^2 = \sum_{i} w_i^2 \,,\ w \in R^n$$

and $\bar{x}$ is the optimal solution of the HASTUS-macro problem. For simplicity of notation we consider for the moment that the pieces of work are defined on periods as in the HASTUS-macro problem. The problem can be easily solved using a heuristic procedure. In fact, if all but one of the decisions vectors y are fixed ($y^{\ell} = \bar{y}^{\ell}$, $\ell \neq r$), the nonlinear objective function of (4) becomes a linear function of the elements of the remaining decision vector, y^r. This function is :

$$(B\bar{x} - \sum_{\ell \neq r} \bar{y}^{\ell})^2 + (e - 2(B\bar{x} - \sum_{\ell \neq r} \bar{y}^{\ell}))y^r$$

where e is a vector of ones and use is made of the equality $(y^r_{ij})^2 = y^r_{ij}$.

Note that the constraints of the problem in the y^r variable vector are those of the shortest path problem. We define

$$V^r(\bar{x},\bar{y}) \cdot (e - 2(B\bar{x} - \sum_{\ell \neq r} \bar{y}^{\ell}))$$

Then, in order to solve (4), we must solve a sequence of shortest path subproblems P_r :

$$P_r \quad : \text{Min } v^r(\bar{x},\bar{y})y^r$$

$$\text{s.t. } E\, y^r = b^r$$

$$y^r \in Y$$

The algorithm MINSQ which finds a suboptimal solution to (4) is described below

MINSQ

1. a) Take any feasible solution $\bar{y}_0$ and evaluate

$$Z^0 = (B\bar{x} - \sum_{\ell} \bar{y}_0^{\ell})^2.$$

b) $k \leftarrow 1$
$\ell \leftarrow 1$
$\bar{y} \leftarrow \bar{y}_{k-1}$

2. a) Solve P_ℓ with $\bar{y}$; let $\bar{y}_k^\ell$ be the solution obtained.

b) $\bar{y}^\ell \leftarrow y_k^\ell$

c) $\ell \leftarrow \ell+1$

d) if $\ell < L$ go to a.

e) evaluate $Z^k = (B\bar{x} - \sum_{\ell} \bar{y}^{\ell})^2$

3. a) If $Z^k = Z^{k-1}$, stop.

b) $k \leftarrow k+1$ go to 2.

Let $\tilde{y}^\ell$ (ℓ=1,.,L) denote the solution obtained for problem (4). This solution satisfies the constraints (iii) of problem (2).

In reality, when this algorithm is applied, the objective function $(B\bar{x}-\sum y^\ell)^2$ uses pieces defined on periods, while the shortest path algorithm uses pieces defined in real time. For the evaluation of the objective function we transform the pieces defined in real time into pieces defined on periods. When the algorithm stops we thus have a partition of each block into pieces of work. The next step consists of building a first feasible solution

THE MATCHING PROBLEM

Most algorithms at this step use an assignment algorithm rather than a matching code to solve the problem of making up a workday out of two pieces of work. To use an assignment algorithm, the pieces of work must be split rather arbitrarily into two sets. We have found experimentally that however these two sets were formed, a matching code always gave better results (between ½% and 1%). However, we found that the matching codes on the market took too much time to optimize large scale problems (~400 pieces, ~20,000 possible links) because several reoptimizations were necessary in the marginal improvement step. Because the matching problem considered here is nearly bipartite, we have developed an approach based on the equivalent flow formulation of the continuous matching problem.

We consider the following matching problem :

$$\text{Min} \sum_{I'} c_{ijkh}\, x_{ijkh}$$

$$\sum_{I'(ij)} x_{mnkh} = 1, \qquad (ij)\varepsilon V \tag{5}$$

$$x_{ijkh} = 0,1$$

V : the set of feasible pieces defined in real time resulting from the partitioning of the blocks.

I' : the set of feasible workdays using pieces from V.

I'(ij) : the set of feasible workdays $(mnkh)\varepsilon I'$ where one of the pieces is (ij).

Note that, in contrast to problem (1), only the x_{ijkh} which are feasible and use pieces of work previously generated by the partitioning algorithm are generated.

Relaxing the integrality of (5) we obtain

$$\text{Min} \sum_{I'} c_{ijkh}\, x_{ijkh}$$

$$\sum_{I'(ij)} x_{mnkh} = 1 \qquad (ij)\in V \qquad (6)$$

$$x_{ijkh} \geq 0$$

It is demonstrated in Appendix 1 that a flow problem equivalent to the L.P. formulation of the matching problem above can be formulated as in Figure 2. Similar results are also presented in Minoux [10].

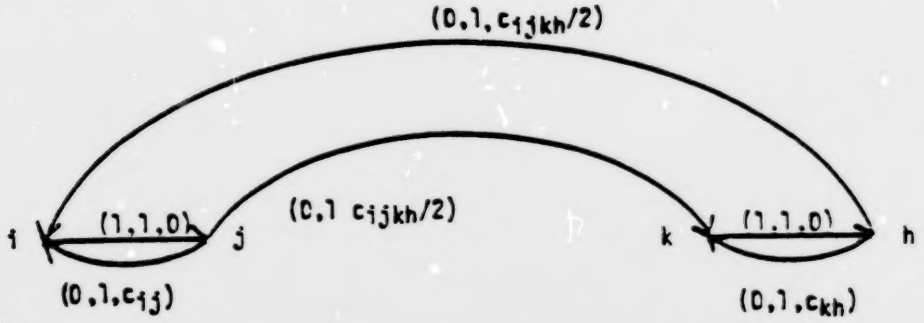


Figure 2

where (ℓ,w,c) means (lower bound, upper bound, cost)

- for each existing piece (i,j) nodes i and j and one directed arc (ij) are created with a minimum and maximum flow of one unit, and a cost of 0. An arc from j to i is also created to allow for the fact that the piece (i,j) may be retained as a tripper with a cost of c_{ij};
- for each feasible workday composed of two pieces (i,j) and (k,h) two arcs are created, one from j to k and one from h to i, both with a cost of $c_{ijkh/2}$.

An equivalent assignment problem could also be generated (see Appendix 1).

If we refer to the value of the arc (ij) as Z_{ij}, an optimal solution of problem 6 can be obtained as

$$x_{ijkh} = \tfrac{1}{2}(z_{jk} + z_{hi})$$

$$x_{ij} = z_{ij}$$

So every cycle containing one or two pieces corresponds to either a tripper, a continuous workday, or a two-piece workday. For every cycle containing an even number of pieces, it can be proved (see Appendix 1) that an equivalent solution can be obtained with two pieces per cycle. Finally a cycle containing an odd number of pieces corresponds to a non-integer solution of the L.P. matching problem (6). Our experience with the approach has revealed that most of the time no odd cycles are obtained. In less than 10% of the cases, there was at most one cycle and odd cycles were present on very rare occasions.

If the final assignment includes an odd cycle, a constraint is generated to eliminate this cycle. This is written as follows :

$$\sum_{(j,k)\in C} z_{jk} + \sum_{(h,i)\in C'} z_{hi} \leq n_c - 1$$

where C : set of arcs of the solution in the odd cycle;

C': set of arcs linking the same pieces in the odd cycle but in the opposite direction;

n_c: number of arcs in the odd cycle.

It is easy to solve the flow problem with one additional constraint using the technique proposed either by Glover et al. [6] or by Masch [9]. Flow algorithms that can handle more than one constraint would be useful but are not available. An alternative method is proposed by Minoux [10] and will be tested in the near future.

On the very rare occasions where more than two odd cycles are present, or when an additional constraint does not eliminate all odd cycles, a piece of work is fixed as a tripper in the odd cycles thus giving a very good approximate solution to the matching problem. When workdays with more than two pieces of work have to be considered, larger pieces of work made up of two smaller pieces are first created before

entering the matching algorithm. Several trials with different ways of forming these larger pieces are considered in order to generate the best possible solution.

After this step, a first feasible solution for the problem has been obtained. In the next step, the solution is marginally improved.

THE MARGINAL IMPROVEMENT ALGORITHM

In the current version of the algorithm, a series of alternative partitions are generated for each block from the information given by HASTUS-macro. Each of these partitions is tried in turn in the matching algorithm. The strategy under development uses the marginal information from the matching problem to generate a better partition of certain blocks in order to obtain a better set of workdays.

The main difficulty is to generate the marginal cost information on the pieces of work. For the pieces already in the network, the marginal cost information obtained from the dual variables of the flow algorithm are not very significant because of the high level of degeneracy of the problem. Moreover these represent only a small portion of all feasible pieces of work.

Initially, the marginal cost is evaluated only for representative pieces such as the pieces defined on periods as in HASTUS-macro. This considerably reduces the number of calculations which have to be carried out. The evaluation of marginal cost is thus carried out by inserting one at a time these representative pieces into the network with all their corresponding arcs linking to pieces already in the network; the reoptimization of the system is carried out only on the new arcs added. The difference in cost of the objective function before and after reoptimization corresponds to an approximate evaluation of the marginal cost of including the given piece in the network.

Each block is chosen in turn to have its partition reevaluated. All feasible pieces of work are created for that given block. In general, the cost corresponding to the marginal cost of the closest representative piece is attributed to each of these feasible pieces and the partition is obtained through a shortest path algorithm as previously. However, for the pieces of work already in the network as part of the previously used partition of the block, we preferred to use a more constant evaluation of marginal cost as follows :

Let c_{ijkh} be the cost of the workday made up of a piece i,j in the block under scrutiny and a piece k,h from another block.

The marginal cost c^*_{ij} will be obtained as follows :

$$c^*_{ij} = c_{ijkh} - c^*_{kh}$$

where c^*_{kh} is the marginal cost of piece k,h as obtained from the representative piece of work.

Each time a new partition is obtained, it replaces the old partition in the network and the matching problem is reoptimized. If the solution is better, the new partition is kept while if the solution is worse, the old partition is reinstalled. A new block is then examined. After a certain number of modifications, or when too many new block partitions are rejected, the marginal cost for the representative pieces are recalculated and the process is pursued until the improvement become insignificant.

We also find it faster to partition two blocks at a time in the early stages of the algorithm before a reoptimization of the problem is carried out. This strategy is also useful for a rapid elimination of trippers. Once a block b has been chosen, a second block to be partitioned is immediately selected as follows :

Among the set of all pieces of work matched to a piece of block b, choose the one with the highest marginal cost which belongs to a block large enough to be partitioned. This block is the second block to be partitioned.

The two blocks are then partitioned independently before a new reoptimization is carried out. Finally, problems still remain with the trippers. The marginal information obtained previously tends to penalize trippers in proportion to their duration, not taking into account the fact that there may be several other trippers at the same time of day. It is obvious that the more similar trippers there are, the more difficult it is to eliminate them. To solve this problem, each tripper is given a fixed initial marginal cost of c_p with a variable cost corresponding to the number of trippers present in the solution at approximately the same time:

$$c_{ij} = c_p + p\, N_{ij}$$

where c_p : is the fixed cost for a tripper

p : a penalty factor

N_{ij} : number of trippers beginning in period i and ending in period j.

The cost of the trippers is modified to the previously defined cost before the evaluation of the marginal cost is carried in the matching problem. This is done so that this calculation affects both the marginal cost of a piece of work that could form a workday with a tripper, and the marginal cost of the representative piece of work that corresponds to a tripper.

These are the main elements of our current research activity aimed at finding better ways to modify the block partitions from the results of the matching problem. The calibration of the parameters and of the strategy is not completed yet but the results obtained to date are quite encouraging.

While the current HASTUS system is attaining impressive results on weekdays where there is a peak to basis service ratio of some importance, (3% to 5% savings), the system's performance was average on week-ends, where there are practically no penalties due to spread and other union

contract rules. In fact, for week-ends, the schedulers were able to improve significantly on the results produced by the HASTUS-micro algorithm using the interactive tools of the HASTUS system [1]. With the marginal improvement algorithm described here, we were able to produce, using the HASTUS-micro algorithm directly, results which were better then or equal to those obtained by the scheduler after many interactive manipulations.

TABLE 1
Cost comparison

Problem	Cost of solution directly obtained from new version of HASTUS-micro[1]	Cost of solution currently produced by scheduler from current version of HASTUS[2]
1	13 579.02	13 564.88
2	9 443.69	9 415.57
3	8 443.13	8 478.39
4	6 456.69	6 463.39
5	13 747.78	13 772.25
6	9 815.78	9 815.83
Total	61 486.09	61 510.31

[1] Some additional improvements could probably be obtained by the scheduler with interactive manipulation.

[2] These correspond to solutions that were in operation recently. They have been produced from HASTUS-micro and improved by a scheduler using the interactive facility of HASTUS.

To conclude this section, we summarize the marginal improvement algorithm that is used :

Module 1 : Evaluate marginal costs for representative pieces :

i) evaluate the cost of trippers and modify accordingly the cost of trippers in the matching problem;

ii) add in turn each representative piece to the matching problem and reoptimize marginally on the new ones, created to obtain their marginal cost;

iii) modify the marginal cost of the pieces used in the current solution so that $c_{ijkh} = \bar{c}_{ij} + \bar{c}_{kh}$.

Module 2 : For each block obtained (or a pair of blocks if the two block option is selected) :

i) generate all feasible pieces of work for that block;

ii) assign to each piece the marginal cost of its representative piece;

iii) solve the shortest path problem.

Module 3 : Reoptimize the matching problem with the new partition(s) obtained :

i) reoptimize first on the new arcs and then on the whole problem;

ii) if the new solution is better, it is kept as the new current solution. If the new solution is worse, the old solution is kept;

iii) determine whether the marginal cost should be recalculated, and then perform Module 1 and/or go to Module 2 to analyse the next block.

Module 4 : After all blocks have been examined, the process may be restarted if necessary or can be terminated.

CONCLUSION

We have described here current development activities to further improve the solution produced by the HASTUS system. When this improvement stage is completed, we intend to fully develop and integrate into the system the work currently pursued by David Scott [12] for producing vehicle schedules that take into account the driver's assignment constraint. Another module for the rostering problem important for European companies will also be developed along the lines described by Carraresi et al. [4] and by Hagberg [7].

APPENDIX 1

The maximal matching problem with costs.

The problem involves an undirected graph $G=(V,E)$ with a set V of nodes and a set E of arcs. According to this notation, the arc joining nodes $i,j \in V$ is written as $e_{ij} \in E$. A subset of E is called a matching if no node V is incident with more than one arc in M. A matching M containing the largest possible number of arcs is defined as a matching of maximal cardinality. Such a matching is considered "perfect" if each node is incident with one arc of the matching.

If the costs on the arcs are denoted by c_{ij}, then the following cost is associated with a matching M :

$$C(M) = \sum_{e_{ij} \in M} c_{ij}$$

The maximal matching problem with costs can be defined by :

$$\text{Min} \quad \sum_{e_{ij} \in E} c_{ij}\, y_{ij} \qquad (1)$$

$$\text{s.a.} \quad \sum_{e_{ij} \in \delta(i)} y_{ij} = 1 \qquad \text{for all } i \in V$$

$$y_{ij} = 0 \text{ or } 1 \qquad \text{for all } e_{ij} \in E$$

where $\delta(i)$ is the set of all arcs incident with node $i \in V$.

If the integrality of variables y_{ij} is relaxed, the following linear problem is obtained

$$\text{Min} \quad \sum_{e_{ij} \in E} c_{ij}\, y_{ij}$$

$$\sum_{e_{ij} \in \delta(i)} y_{ij} = 1 \qquad \text{for all } i \in V \qquad (2)$$

$$y_{ij} \geq 0 \qquad \text{for all } e_{ij} \in E$$

Consider the following flow problem :

$$\text{Min} \sum_{(i,j')\in A} d_{ij}\, x_{ij'} \qquad (3)$$

$$\text{s.a.} \sum_{j'\in I(i)} x_{ij'} = 1 \quad \text{for all } i \in V_1$$

$$-\sum_{i\in I'(j')} x_{ij'} = -1 \quad \text{for all } j' \in V_2$$

$$x_{ij'} \geq 0 \quad \text{for all } (i,j') \in A$$

where V_1 : set of source nodes

V_2 : set of sink nodes

$I(i)$: set of nodes $\in V_2$ linked to the node $i \in V_1$

$I'(j')$: set of nodes $\in V_1$ linked to the node $j' \in V_2$.

In addition, we have :

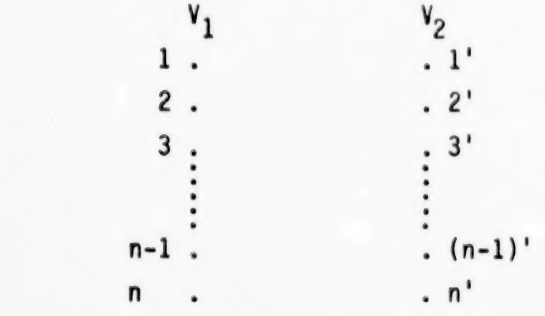
- $j' \in I(i) \leftrightarrow i \in I'(j')$

  i.e. an arc links nodes i and j' if and only if there exists an arc linking j to i';

- $d_{ij'} = d_{ji'} = c_{ij}/2$ for all $(i,j') \in A$

$|V_1| = |V_2| = |V|$

$(i,j') \in A$ if and only if $e_{ij} \in E$



In problem (3), the fact that a piece of i could make a workday with a piece of work j could be represented by either Fig. 3a or Fig. 3b.

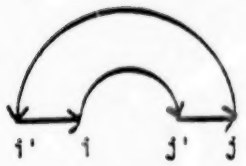


Figure 3a
A flow of one is required on arc (i'i) and (j'j)

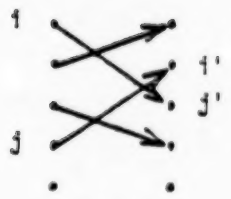


Figure 3b
A flow of one enters i and j and must exit from i' and j'

Figure 3

Representation of problem (3)

Lemma 1 : By solving optimally problem 3, we obtain an optimal solution for problem 2.

i) given $\bar{x}_{ij'}$ an optimal solution to the problem 3, $\bar{y}_{ij}=(\bar{x}_{ij'}+\bar{x}_{ji'})/2$ is a feasible solution of problem 2 because

$$\tfrac{1}{2}\Big(\sum_{j'\in\Gamma(i)} \bar{x}_{ij'} + \sum_{j\in\Gamma'(i')} \bar{x}_{ji'}\Big) = \sum_{j'\in\Gamma(i)} \sum_{j\in\Gamma'(i')} \tfrac{1}{2}(\bar{x}_{ij'}+\bar{x}_{ji'})$$

$$= \sum_{e_{ij}\in\delta(i)} y_{ij} = 1$$

ii) Given $\bar{\alpha}_i$ and $\bar{\alpha}_{i'}$ the optimal solution of the dual of problem (3)

$$\text{Max} \sum_{i\in V_1} \alpha_i - \sum_{i'\in V_2} \alpha_{i'}$$

$$\text{s.t.} \quad \alpha_i - \alpha_{j'} \leq d_{ij'} \qquad \text{for all } (i,j')\in A$$

then $\bar{\gamma}_i = \bar{\alpha}_i - \bar{\alpha}_{i'}$ is a feasible solution of the dual of problem (2).

$$\text{Max} \sum_{i\in V} \gamma_i$$

$$\text{s.t.} \quad \bar{\gamma}_i + \bar{\gamma}_j \leq c_{ij} \qquad \text{for all } (i,j)\in E$$

because $\bar{\alpha}_i - \bar{\alpha}_{j'} \leq d_{ij'}$

$\bar{\alpha}_j - \bar{\alpha}_{i'} \leq d_{ji'}$

so $(\bar{\alpha}_i - \bar{\alpha}_i') + (\bar{\alpha}_j - \bar{\alpha}_j') \leq d_{ij'} + d_{ji'}$

and $\bar{\gamma}_i + \bar{\gamma}_j \leq c_{ij}$

iii) $\sum_{(ij')\in A} d_{ij'} \bar{x}_{ij'} = \sum_{i\in V_1} \bar{\alpha}_i - \sum_{i'\in V_2} \bar{\alpha}_{i'}$

because of the optimality of the solution to problem 3 so

$$\sum_{e_{ij}\in E} c_{ij} \bar{y}_{ij} = \sum_{i\in V} \bar{\gamma}_i$$

which demonstrates that $\bar{y}_{ij}$ and $\bar{\gamma}_i$ are optimal solutions of the primal and dual problem of problem (2).

To obtain an integer solution for problem (2) from problem (3) both arcs (ij') and (ji') must be in the flow solution of problem (3) (i.e $x_{ij'} = x_{ji'} = 1$).

This is obviously not always the case. An (ij') might be in the solution of problem (3) ($x_{ij'} = 1$) while there might be a cycle from node j to node i' that contains more than one arc, or in other words, the cycle from i to i' might contain more than two arcs as for example :

$$(ij')\ (jj_1')\ (j_1j_2')\ (j_2i')$$

The number of arcs from i to i' could be odd or even. In the following lemma we demonstrate that as long as there are no cycles of odd length in problem (3) we can find an integer optimal solution for problem (2).

Lemma 2 : If all cycles in problem (3) are even there exists an integer optimal solution for problem (2).

If all cycles have length 2 in the optimal solution of problem (3), the solution of problem (2) is integer by construction.

Suppose we have a cycle of even length greater than 2 in an optimal solution of problem (3)

$(i_1,i'_2)\ (i_2,i'_3)\ \ldots\ (i_{\ell-1},i'_\ell),\ (i_\ell,i'_1)$

The cost of this solution is

$$d_{i_1,i'_2} + d_{i_2i'_3} + \ldots + d_{i_{\ell-1}i'_\ell} + d_{i_\ell i'_1}$$

Because the length of the cycle is even let :

$$D_1 = d_{i_1i'_2} + d_{i_3i'_4} + \ldots + d_{i_{\ell-1}i'_\ell}$$

$$D_2 = d_{i_2i'_3} + d_{i_4i'_5} + \ldots + d_{i_\ell i'_1}$$

Suppose $D_1 < D_2$, we can modify the solution as follows :

$x_{i_1i'_2} = x_{i_3i'_4} = \ldots = x_{i_{\ell-1}i'_\ell} = 1$ as in the solution and

let

$$x_{i_2i'_3} = x_{i_4i'_5} = \ldots = x_{i_\ell i'_1} = 0$$

and $x_{i_2i'_1} = x_{i_4i'_3} = \ldots = x_{i_\ell i_{\ell-1}} = 1$.

This solution is obviously feasible and its cost is $2D_1$ instead of D_1+D_2. The difference is $D_1-D_2<0$ which contradicts the fact that the previous solution is optimal for problem (3). By a symmetric argument we can demonstrate that $D_2<D_1$ leads to the same construction which shows that $D_1=D_2$ and that the alternative solution proposed is also an optimal solution to problem (3). All cycles of even length greater than 2 can thus be reduced to cycles of length 2 in this manner and an integer optimal solution for problem (2) can be obtained.

REFERENCES

[1] BLAIS, J.-Y. and ROUSSEAU, J.-M., HASTUS system : an interactive system for vehicle and crew scheduling. Preprints of the Third International Workshop on Vehicle and Crew Scheduling, Université de Montréal, June 27-30, 1983.

[2] BLAIS, J.-Y. and ROUSSEAU, J.-M., HASTUS : A model for the economic evaluation of drivers' collective agreements in transit companies, INFOR 20 (1982) 3-15.

[3] BODIN, L., GOLDEN, B., ASSAD A. and BALL, M., Routing and scheduling of vehicles and crews - The state of the art. Computer and Operations Research 10, 2, Special Issue, (1983).

[4] CARRARESI, P., GALLO, G. and ROUSSEAU, J.-M., Relaxation approaches to large-scale bus drivers, scheduling problems. Transportation Research B 16B (1982) 383-397.

[5] CARRARESI, P., GALLO, G., BELLETTI, R. and DAVINI, A., BDROP - A package for the bus drivers' rostering problem. Preprints of the Third International Workshop on Transit Vehicle and Crew Scheduling, Université de Montréal, June 27-30, 1983.

[6] GLOVER, F., KARNEY, D., KLINGMAN, D. and RUSSEL, R., Solving singly constrained transshipment problems. Transportation Science 12 (1978) 277-297.

[7] HAGBERG, B., An assignment approach to the rostering problem : an application to taxi vehicles. Preprints of the Third International Workshop on Transit Vehicle and Crew Scheduling, Université de Montréal, June 27-30, 1983.

[8] LESSARD, R., ROUSSEAU, J.-M. and DUPUIS, D., HASTUS I : A Mathematical Programming Approach to the Bus Driver Scheduling Problem. Computer Scheduling of Public Transport, Urban Passenger Vehicle and Crew Scheduling, Wren, A., ed., North-Holland Publ. Co. (1981) 255-267.

[9] MASCH, V.V., The cyclic method of solving the transshipment problem with an additional linear constraint. Networks 10 (1980) 17-31.

[10] MINOUX, M., A new polynomial cutting-plane algorithm for maximum weight matchings in general graphs. Publication #302, Centre de recherche sur les transports, Université de Montréal (1983). Paper presented at the C.R.T., June 1983.

[11] R.A.T.P., Conditions de travail des machinistes européens. Enquête 1978, RP-2. Paris (1979).

[12] SCOTT, D., A large-scale linear programming approach in the public transport scheduling and costing problems. Preprints of the Third International Workshop on Transit Vehicle and Crew Scheduling, Université de Montréal, June 27-30, 1983.

[13] SHARP, G.P., Constraints for scheduling operators for urban public transit systems. Preprints of the Workshop on Automated Techniques for Scheduling of Vehicle Operators for Urban Public Transportation Services, Bodin L. and Bergman, D., eds., Chicago, Ill., U.S.A., April 1975.

[14] *** Preprints of the Workshop on Automated Techniques for Scheduling of Vehicle Operators for Urban Public Transportation Services, Bodin, L. and Bergman, D., eds., Chicago, Ill., U.S.A., April 1975.

[15] *** Computer Scheduling of Public Transport, Urban Passenger Vehicle and Crew Scheduling, Wren, A., ed., North-Holland Publ. Co. (1981).

[16] *** Preprints of the Third International Workshop on Transit Vehicle and Crew Scheduling, Université de Montréal, June 27-30, 1983. Forthcoming as Proceedings Volume of the Workshop to be published by North-Holland.